DOI:
10.1109/TCOMM.2020.2973566

Document Version
Peer reviewed version

Distributed Mirror-Prox Optimization for Multi-Access Edge Computing

Zhenfeng Sun, *Student Member, IEEE,* and Mohammad Reza Nakhai, *Senior Member, IEEE*

***Abstract*—In this paper, we address the problem of overall delay minimization in a cellular multi-access edge computing (MEC) network, where servers with limited computing and storage resources are co-located with a base station (BS) to execute the computation tasks offloaded by the users. We formulate this problem as a distributed mirror prox (DMP) optimization at individual users and the MEC servers local controller (LC) over a long-time horizon and develop an online algorithm to ensure queue stability of the overall network in the long run. Both, the cost and the constraint functions are time-varying with unknown statistics. We evaluate the performance of the proposed algorithm using two performance metrics: the dynamic regret to assess the closeness of the achievable cost against the dynamic optimal value; and the aggregate violation to measure the asymptotic satisfaction of the constraints. Given the optimum hindsight variation is sub-linear, we prove that both of the dynamic regret and the aggregate violation are sub-linear in the long run. The simulation results confirm the superiority of the proposed DMP algorithm over the stochastic dual gradient in terms of delay minimization, dynamic regret, aggregate violation and energy efficiency in battery-powered user devices.**

***Index Terms*—Distributed Mirror prox, dynamic regret, projected dual gradient, aggregate violation of constraints, multi-access edge computing.**

I. Introduction

The emerging architecture of multi-access edge computing (MEC) can extend cloud computing to the edge of the network to enable ultra-low latency execution of computation-intensive applications at various smart devices (SDs) like smartphones and tablet-PCs with limited computation and energy resources [1], [2]. These applications include gesture and face recognition, mixed reality, 3D modelling, online gaming and internet of things (IoT) networks for smart homes and cities [3]. Billions of SDs are estimated to be deployed in near future, which are in need of MEC to reduce delay and energy consumption [4]. However, devising online control mechanisms to achieve this goal is challenging due to the natural exposure of SDs to highly dynamic environment with unpredictable amount of arriving tasks, different CPU-cycle frequency of SDs, varying transmission power, transmission rate between SDs and MEC servers and different computing intensities required by various applications.

Manuscript received March 13, 2019; revised August 16, 2019; accepted February 03, 2020. The editor coordinating the review of this paper and approving it for publication was Prof. Ting He. (Corresponding author: Mohammad Reza Nakhai.)

Z. Sun and M. R. Nakhai are with the Department of Engineering, Centre for Telecommunications Research, Kings College London, London WC2R 2LS, U.K. (e-mail: zhenfeng.sun@kcl.ac.uk; e-mail: reza.nakhai@kcl.ac.uk).

A. Related work

A few recent works have considered the resource allocation in MEC system [5]–[7]. Authors in [6] have proposed a stochastic optimization approach to schedule the computation tasks based on the queueing state, using Markov method. In [5], three sub-problems of minimizing power consumption under the constrained delay in MEC-cloud system by workload allocation is formulated and solved by three sub-algorithms, respectively. Authors in [7] have formulated a single optimization problem for computation offloading and resource allocation in a cellular network with MEC and caching capability and solved it in a distributed manner using the alternating direction method of multipliers (ADMM). Although, the resulting decisions in [5] and [7] are optimal over the current decision making time-interval, their optimality over a long time-horizon cannot be ensured due to time-varying task arrivals, instantaneous computation intensities and the random nature of wireless links.

Authors in [8] leverage the saddle point method to solve the distributed online learning networked problem and uses the static regret, which is based on the best fixed decision from the feasible area in hindsight as the benchmark of the performance. However, due to the time-varying nature of the underlying cost and constraint functions, the static regret may be a too optimistic measure in capturing the dynamic nature of a realistic environment [9], as compared to the dynamic regret [10], [11]. The recent works [10] and [11] although extend the static regret to the dynamic regret, their constraint functions are set as time-invariant which cannot be violated instantaneously. For the edge computing, we need to make online decisions based on the time-varying cost functions and the time-varying constraints to deal with the dynamic environment (e.g. the computing resources and the wireless channel).

B. Contributions

In this paper, we focus on the efficient control of the amount of data to be processed locally at the SDs, the amount of data to be offloaded to the MEC servers and the computing process at the MEC servers, such that the data processing delay is minimized and the queue stability across the entire network is assured in the long run. We formulate these tasks as a distributed mirror-prox optimisation problem across the SDs and the servers' LC at the BSs under a stochastic setting of the network environment. We then develop the DMP algorithm that solves this problem distributively using a single broadcast of MEC servers' state to the SDs at the end of each time-slot.

Since, the designs based on the static regret may lead to an unrealistic result in practical scenarios [9], we use the dynamic regret and the aggregate violation as more realistic metrics to evaluate the performance of the proposed algorithm. We analytically prove that the proposed DMP algorithm achieves a sub-linear performance for both of the dynamic regret and the aggregate violation of constraints in the long run, given a sub-linear accumulated hindsight optimum variation. Our simulation results also confirm that the proposed DMP algorithm achieves lower time-averaged network delay as well as lower battery energy consumption at the SDs than its counterpart, the stochastic dual gradient. The lower energy consumption also implies longer battery life in the battery-powered SDs.

Throughout the paper, a, $\mathbf{a}$ and $\mathbf{A}$ denote a scalar a, a column vector $\mathbf{a}$ and a matrix $\mathbf{A}$, respectively. $\mathcal{M}$ represents the index set. $\mathbf{I}$ and $\mathbf{1}$ are the identity matrix and the all-ones vector, respectively. The notations $(\cdot)^\top$ and $\|\cdot\|$ indicate the vector or matrix transposition and l_2(Euclidean) norm of a vector, respectively. The notation $[\mathbf{a} \vdots \mathbf{b} \vdots \mathbf{c}]$ indicates concatenating the vector $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{c}$ as one vector. The inequality for vectors, e.g., $\mathbf{a} \geq \mathbf{0}$, is defined entry-wise. The notation $\mathcal{P}_{\mathcal{X}}\{\cdot\}$ is denoted as a vector projected to a specified area $\mathcal{X}$. The notation $O(\cdot)$ and $o(\cdot)$ represent big O and little o, respectively.

The rest of the paper is organized as follows. Section II introduces the system model from SDs to the MEC servers. In Section III, the online optimization problem and two performance metrics, i.e., dynamic regret and aggregated violation, are developed. In section IV, the proposed DMP algorithm and the performance analysis are presented. Simulation results will be shown in Section V. Finally, the conclusion is given in Section VI.

II. System Model

Fig. 1 shows a multi-access edge computing (MEC) system where M smart devices (SDs) offload part of their computing tasks to the MEC platform comprising of N servers and a local controller (LC). The MEC platform is attached to the BS and executes offloaded computation-intensive tasks by the SDs. Let the sets $\mathcal{M} = \{1, ..., M\}$, $\mathcal{N} = \{1, ..., N\}$ and $\mathcal{T} = \{1, ..., T\}$, respectively, denote the indices of SDs, the indices of MEC servers and the discrete time-slots. Note that, T is the time horizon during which sequential decision-makings take place every τ unit of time, i.e., the duration of a time-slot. The amount of data received by the mth SD at time-slot t is b_t^m in bits. The SDs share a total bandwidth of W Hz using the frequency division multiple access (FDMA) scheme.

The amount of locally processed data at the mth SD is denoted by d_t^m (bits). The computation intensity of the mth SD at time-slot t, which can be considerably variant among the SDs due to the different types of received tasks, is denoted by w_t^m (cycles per bit). The local processing delay at time-slot t in the mth SD can be expressed as $f_t^m(d_t^m) = \frac{d_t^m w_t^m}{\rho_t^m}$, where ρ_t^m in cycles/sec indicates the CPU-cycle frequency of the mth SD at time-slot t. We also denote the upper-bound of ρ_t^m as ρ_u^m.

TABLE I
SUMMERY OF VARIABLES

Notation	Description
$\mathcal{T}, T, t$	Set, number and index of time-slots
$\mathcal{M}, M, m$	Set, number and index of SDs
$\mathcal{N}, N, n$	Set, number and index of MEC servers
d_t^m	Amount of data processed locally at the mth SD at time-slot t
a_t^{mn}	Amount of offloaded data to the nth server at time-slot t
z_t^n	Amount of data processed by the nth MEC server at time-slot t
b_t^m	Amount of data received by mth SD at time-slot t
w_t^m	Computation intensity of the received task at the mth SD at time-slot t
ρ_t^m	CPU-cycle frequency of the mth SD at time-slot t
ρ_t^n	CPU-cycle frequency of the nth MEC server at time-slot t
κ_m, κ_n	Effective switched capacitance of CPU at the mth SD and at the nth MEC server
r_t^m	Transmission rate between mth SD and the MEC servers
W	Channel bandwidth
σ^2	Gaussian noise
$p_{tr,t}^m$	Transmission power of the mth SD
h_t^m	Channel gain between mth SD and the MEC servers
α, μ	Primal and dual step sizes
$\mathcal{V}_{\mathbf{x}_t^*}^T$	Accumulated hindsight optimum variation

The processing delay of the offloaded tasks by the SDs to the MEC servers is the sum of transmission delay through the intermediate wireless channel and the computation delay at the MEC servers. Let us denote the amount of offloaded data to the nth MEC server by the mth SD at time-slot t as a_t^{mn} (bits), the CPU-cycle frequency of the nth MEC server at time-slot t as ρ_t^n (cycles/sec), which cannot exceed its maximum value ρ_u (cycles/sec). Let the small scale fading, e.g., [12], power gain between the mth SD and the BS be denoted by γ_t^m at time-slot t, which is exponentially distributed with mean $\frac{1}{\xi}$, i.e., $\gamma_t^m \sim \text{Exp}(\xi)$. Hence, the corresponding channel power gain can be expressed as $h_t^m = \gamma_t^m q_0 (l_0/l_t)^\theta$, where q_0 is the path-loss constant, l_0 is the reference distance, l_t is the distance between the mth SD and the BS in the tth time-slot, and θ is the path-loss exponent. Since FDMA transmission is used, the simultaneous transmissions of SDs do not interfere with one another and the transmission rate of r_t^m (bits) for computation offloading by the mth SD at time-slot t can be expressed according to the Shannon-Hartley formula [13] as

$$r_t^m = \phi_t^m W \log_2 \left(1 + \frac{p_{tr,t}^m h_t^m}{\phi_t^m W N_0}\right), \tag{1}$$

where W is channel bandwidth, N_0 is the noise power spectral density at the BS, $p_{tr,t}^m$ is the transmission power generated by the mth SD, h_t^m is the channel power gain from mth SD to the MEC server, ϕ_t^m is the proportion of bandwidth allocated to the mth SD. The bandwidth allocation is assumed as equal distribution, i.e., $\phi_t^m = 1/M, m \in \mathcal{M}$. Hence, the transmission delay of task offloading at time-slot t can be expressed as $f_{tr,t}^{mn}(a_t^{mn}) = \frac{a_t^{mn}}{r_t^m}$.

Similarly, the computation delay at MEC servers are given as $f_{ex,t}^n(z_t^n) = \frac{z_t^n w_t^n}{\rho_t^n}$, where w_t^n is the computation intensity for the arriving tasks at the nth MEC server. Thus, the overall

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2020.2973566, IEEE Transactions on Communications

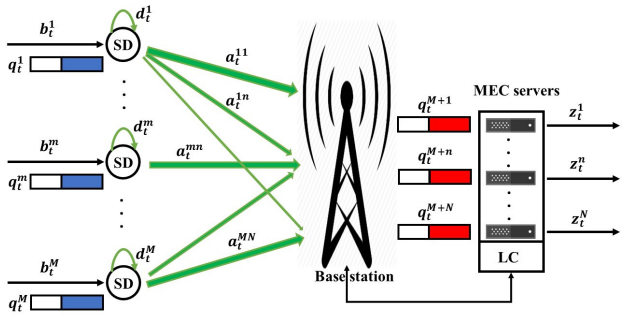


Fig. 1. A graph of online resource allocation between SDs and the MEC servers. Per time-slot t, the mth SD receives the arriving workload b_t^m and stores them in the queue q_t^m. Then, the workload in the queue q_t^m will be allocated to locally process d_t^m and offload to the MEC servers a_t^{mn}, $\forall n \in \mathcal{N}$. The nth MEC server receives the amount of workload $\sum_{m=1}^{M} a_t^{mn}$ and stores them in the queue q_t^{M+n}. Finally, the nth MEC server executes the amount of z_t^n out of the workload in its queue q_t^{M+n}, which are recorded by local controller (LC).

delay for tasks offloaded at time-slot t, i.e, $\psi_t(d_t^m, a_t^{mn}, z_t^n) = \sum_{m=1}^{M} f_t^m(d_t^m) + \sum_{m=1}^{M}\sum_{n=1}^{N} f_{tr,t}^{mn}(a_t^{mn}) + \sum_{n=1}^{N} f_{ex,t}^n(z_t^n)$, is given by

$$\psi_t(d_t^m, a_t^{mn}, z_t^n) = \sum_{m=1}^{M} \frac{d_t^m w_t^m}{\rho_t^m} + \sum_{m=1}^{M}\sum_{n=1}^{N} \frac{a_t^{mn}}{r_t^m} + \sum_{n=1}^{N} \frac{z_t^n w_t^n}{\rho_t^n}. \quad (2)$$

To describe the energy consumption results of the SD and MEC servers in the case of controlling data traffic, the energy consumption for mth SD is formulated as

$$E_{\mathrm{SD},t}^m = \underbrace{\kappa_m d_t^m w_t^m (\rho_t^m)^2}_{\text{local consumption}} + \underbrace{p_{tr,t}^m a_t^{mn}/r_t^m}_{\text{transmission consumption}}, \quad (3)$$

where κ_m is the effective switched capacitance of the CPU at the mth SD. Then, the energy consumption for nth MEC server is given by

$$E_{\mathrm{MEC},t}^n = \kappa_n z_t^n w_t^n (\rho_t^n)^2, \quad (4)$$

where κ_n is the effective switched capacitance of the CPU at the nth MEC server.

III. Problem Formulation

Let $\mathbf{d}_t = [d_t^1, \cdots, d_t^M]^\top$ and $\mathbf{a}_t = [a_t^{11}, \cdots, a_t^{MN}]^\top$ stack, respectively, the amounts of data scheduled to be processed locally at SDs and the amount of data to be offloaded to MEC servers. Moreover, $\mathbf{z}_t = [z_t^1, \cdots, z_t^N]^\top$ denotes the amount of executed data by the MEC server. Let d_u, a_u and z_u indicate the upper limits for the computing capability of SDs, the transmission capacity and the computing capability of MEC servers, respectively, i.e., $d_t^m \le d_u$, $a_t^{mn} \le a_u$ and $z_t^n \le z_u, \forall m, n, t$. Let us define $\mathbf{x}_t = \left[\mathbf{d}_t^\top \vdots \mathbf{a}_t^\top \vdots \mathbf{z}_t^\top\right]^\top$ and $\mathbf{x}_u$ that stacks the upper-bound values of d_t^m, a_t^{mn} and z_t^n. Thus, the feasible region of $\mathbf{x}_t$ is given by $\mathcal{X} = \{\mathbf{0} \preceq \mathbf{x}_t \preceq \mathbf{x}_u\}$, which is assumed to be convex.

In the sequel, we formulate an online optimization problem to make online decisions on $\mathbf{x}_t$ for offloading policy. Let us define $\mathbf{c}_t = \left[\mathbf{c}_{l,t}^\top \vdots \mathbf{c}_{r,t}^\top \vdots \mathbf{c}_{m,t}^\top\right]^\top = \left[j_t^1, \cdots, j_t^M \vdots k_t^{11}, \cdots, k_t^{MN} \vdots u_t^1, \cdots, u_t^N\right]^\top$, where $j_t^m = \frac{w_t^m}{\rho_t^m}, m = 1, ..., M$, $k_t^{mn} = \frac{1}{r_t^m}, mn = 1, ..., MN$ and $u_t^n = \frac{w_t^n}{\rho_t^n}, n = 1, ..., N$. Then, the overall network delay cost at time-slot t can be written as

$$\psi_t(\mathbf{x}_t) = \mathbf{c}_t^\top \mathbf{x}_t. \quad (5)$$

Let $\mathbf{q}_t = [q_t^1 \cdots q_t^M \vdots q_t^{M+1} \cdots q_t^{M+N}]^\top$, stack the unserved workload at the queues of the SDs, denoted as q_t^m, where $m \in \mathcal{M}$, and the unserved workload at the servers, denoted as q_t^{M+n}, where $n \in \mathcal{N}$. Then the queue dynamic over time-slots can be expressed as

$$\mathbf{q}_{t+1} = \mathcal{P}_{\mathbb{R}_+^{M+N}}\left\{\mathbf{q}_t + \mathbf{b}_t + \mathbf{A}\mathbf{x}_t\right\}, \quad (6)$$

where $\mathcal{P}_{\mathbb{R}_+}\{.\}$ indicates the projection of the argument vector on the positive real quadrant, $\mathbf{b}_t = [b_t^1, \cdots, b_t^M, 0, \cdots, 0]^\top$ and $\mathbf{A}$ is the incidence matrix indicating a mathematical model for the network in Fig. 1 in terms of the flow of information from the entry points at the SDs, through the wireless links between the SDs and the MEC servers and the outgoing served information from the servers. Let $\mathcal{S} = \{1, .., M\} \cup \{(m, n), \forall m \in M, \forall n \in N\} \cup \{1, ..., N\}$ denote all the links pointing back to the SDs, the wireless links connecting the SDs to the the MEC servers and the outgoing virtual links from the MEC servers, in Fig. 1. Let also $\mathcal{I} = \{1, ..., M\} \cup \{1, ..., N\}$ denote the set of all edge nodes, i.e., the SDs and the MEC servers, in Fig. 1. Then, the $(M + N) \times (M + MN + N)$ incidence matrix $\mathbf{A}$ with the (i, s)th entry can be expressed such that $\mathbf{A}_{(i,s)} = 1$ if the link s enters the node i, $\mathbf{A}_{(i,s)} = -1$ if the link s leaves the node i and 0 otherwise, where $i \in \mathcal{I}$ is the index of nodes and $s \in \mathcal{S}$ is the index of links. Thus, the incidence matrix for Fig. 1 can be formulated block-wise, as

$$\mathbf{A} = \left[\begin{array}{c:c:c} -\mathbf{I}_{M\times M} & \mathbf{B}_{M\times MN} & \mathbf{0}_{M\times N} \\ \hdashline \mathbf{0}_{N\times M} & \mathbf{G}_{N\times MN} & -\mathbf{I}_{N\times N} \end{array}\right] = \left[\begin{array}{c:c:c} \underbrace{-\mathbf{I}_{M\times M}}_{\mathbf{A}_l} & \underbrace{\mathbf{B}_{M\times MN}}_{\mathbf{A}_r} & \underbrace{\mathbf{0}_{M\times N}}_{\mathbf{A}_m} \\ \hdashline \underbrace{\mathbf{0}_{N\times M}}_{\mathbf{A}_l} & \underbrace{\mathbf{G}_{N\times MN}}_{\mathbf{A}_r} & \underbrace{-\mathbf{I}_{N\times N}}_{\mathbf{A}_m} \end{array}\right], \quad (7)$$

where $\mathbf{I}_{M\times M}$ is a $M \times M$ identity matrix, $\mathbf{0}_{N\times M}$ is a $N \times M$ matrix with all zeros entries, $\mathbf{1}_{N\times 1}$ is a $N \times 1$ vector with all one elements, $\mathbf{B}_{M\times MN} = \mathrm{BLKdiag}[-\mathbf{1}_{N\times 1}^\top]$ is $M \times MN$ block diagonal matrix with $-\mathbf{1}_{N\times 1}^\top$ entries as the block diagonals and $\mathbf{G}_{N\times MN} = [\underbrace{\mathbf{I}_{N\times N} \vdots \cdots \vdots \mathbf{I}_{N\times N}}_{M}]$ is a $N \times MN$ matrix of M concatenated $N \times N$ identity matrices. An example of the incidence matrix for a simple data flow structure is shown in Appendix A.

In order to assure that all arrived data are served by the end of time horizon T, we should have $\mathbf{q}_{T+1} = \mathbf{0}$ for an initial queue state of $\mathbf{q}_1 = \mathbf{0}$. Furthermore, (6) implies the recursion $\mathbf{q}_{t+1} \succeq \mathbf{q}_t + \mathbf{b}_t + \mathbf{A}\mathbf{x}_t$ for any t, e.g., one can write $\mathbf{b}_T + \mathbf{A}\mathbf{x}_T \preceq \mathbf{q}_{T+1} - \mathbf{q}_T$ for $t = T + 1$. Hence, by successive application of this recursion in the downwards direction for $t = T$, we can equivalently express (6) as $\sum_{t=1}^{T}(\mathbf{b}_t + \mathbf{A}\mathbf{x}_t) \preceq \mathbf{q}_{T+1} - \mathbf{q}_1 \preceq \mathbf{0}$, which assures that all the arrived data workload at SDs are

served by the end of time horizon T. Next, we formulate an online optimization problem over the feasible convex set $\mathcal{X}$ as

$$\min_{\mathbf{x}_t \in \mathcal{X}, \forall t} \sum_{t=1}^{T} \psi_t(\mathbf{x}_t) = \sum_{t=1}^{T} \mathbf{c}_t^\top \mathbf{x}_t$$
$$\text{s. t } \sum_{t=1}^{T} \mathbf{g}_t(\mathbf{x}_t) = \sum_{t=1}^{T} (\mathbf{b}_t + \mathbf{A}\mathbf{x}_t) \preceq \mathbf{0}, \tag{8}$$

where the objective and the constraint are time-varying functions over the time horizon T. At each slot t, a learner chooses a solution $\mathbf{x}_t \in \mathcal{X} \subseteq \mathbb{R}^{M+MN+N}$ (or a feasible action $\mathbf{x}_t$) and applies as the current decision at the beginning of time-slot t. Then, the random nature of the network responds to that action by revealing a cost function $\psi_t(.) : \mathbb{R}^{M+MN+N} \rightarrow \mathbb{R}$ and introducing a constraint function $\mathbf{g}_t(.) : \mathbb{R}^{M+MN+N} \rightarrow \mathbb{R}^{M+N}$, which enforces queue stability in the entire network. However, due to the unknown and time-varying parameters, i.e., the incoming workload b_t^m, the computation intensity w_t^m, CPU-cycle frequency of the SD ρ_t^m and channel fading effects γ_t^m, the constraint function $\mathbf{g}_t(\mathbf{x}_t)$ is not known at the beginning of time-slot t when the allocation decision of $\mathbf{x}_t$ is made. This may result in $\mathbf{g}_t(\mathbf{x}_t) \succeq \mathbf{0}$ and $\mathbf{g}_{t+1}(\mathbf{x}_{t+1}) \preceq \mathbf{0}$, which is tolerable. Because the aggregate constraint $\sum_{t=1}^{T} \mathbf{g}_t(\mathbf{x}_t)$ allows adaptation of online decisions to the environment dynamics when optimizing the per time-slot problem (11), where restricting $\mathbf{x}_t$ to satisfy $\mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}$ implies $\sum_{t=1}^{T} \mathbf{g}_t(\mathbf{x}_t) \preceq \mathbf{0}$. In order to quantify the aggregate violation of constraints, let us define

$$\mathrm{V}_T = \left\| \mathcal{P}_{\mathbb{R}_+^{M+N}} \left\{ \sum_{t=1}^{T} \mathbf{g}_t(\mathbf{x}_t) \right\} \right\|, \tag{9}$$

where V_T is the aggregate violation over time horizon T. Furthermore, the objective function is a function of random quantities, i.e., computational intensity, the CPU capability and channel condition, which are unknown at the beginning of time-slot t to the decision maker and will be revealed at the end of the time-slot duration. Hence, to quantify the performance of the solution to problem (8) with respect to the *dynamic optimum* solution, we define a dynamic regret R_T over time horizon T as

$$\mathrm{R}_T = \sum_{t=1}^{T} \psi_t(\mathbf{x}_t) - \sum_{t=1}^{T} \psi_t(\mathbf{x}_t^*), \tag{10}$$

where $\mathbf{x}_t^* = \arg\min_{\mathbf{x} \in \mathcal{X}} \psi_t(\mathbf{x})$ s. t $\mathbf{g}_t(\mathbf{x}) \preceq \mathbf{0}$, $\forall t \in \mathcal{T}$ are the *dynamic optimum* solutions per time-slot, while $\{\mathbf{x}_t\}_{t=1}^T$ are the actual decisions for the online strategies over the time-slots.

IV. Online Optimization for Computation Offloading

Solving the problem in (8) in its original form is not tractable, because the decisions at each time-slot t requires the information of future time-slots over the entire time horizon T. To overcome this problem, in this section, we reformulate the problem in (8) as a set of optimization problems over the time-slots $t \in \mathcal{T}$ and introduce an algorithm that sequentially solves these problems, such that that the aggregate violation V_T and the dynamic regret R_T defined in (9) and (10), respectively, are both sub-linear with respect to T.

A. Algorithm development

Let the per time-slot problem be expressed as

$$\min_{\mathbf{x}_t \in \mathcal{X}, \forall t} \psi_t(\mathbf{x}_t) = \mathbf{c}_t^\top \mathbf{x}_t$$
$$\text{s. t} \qquad \mathbf{g}_t(\mathbf{x}_t) = \mathbf{b}_t + \mathbf{A}\mathbf{x}_t \preceq \mathbf{0}. \tag{11}$$

The problem in (11) is convex, because the cost function is a linear function with respect to $\mathbf{x}_t$, and the constraint function is affine with respect to $\mathbf{x}_t$. Both cost function and constraint function are only revealed at the end of each time-slot. Hence, $\psi_t(\mathbf{x}_t)$ and $\mathbf{g}_t(\mathbf{x}_t)$ are unknown at the beginning of each time-slot t, which could be completely different from the previous ones, $\psi_{t-1}(\mathbf{x}_{t-1})$ and $\mathbf{g}_{t-1}(\mathbf{x}_{t-1})$. Solution to problem (11) can be found by solving a max-min problem, expressed as

$$\max_{\boldsymbol{\lambda} \in \mathbb{R}_+^{M+N}} \min_{\mathbf{x} \in \mathcal{X}} \mathcal{L}_t^r(\mathbf{x}, \boldsymbol{\lambda}), \tag{12}$$

where

$$\mathcal{L}_t^r(\mathbf{x}, \boldsymbol{\lambda}) = \psi_t(\mathbf{x}) + \boldsymbol{\lambda}^\top \mathbf{g}_t(\mathbf{x}) - \frac{\delta\alpha\|\boldsymbol{\lambda}\|^2}{2} \tag{13}$$

is the regularised Lagrange function of (11) at time-slot t. In (13), $\boldsymbol{\lambda} \in \mathbb{R}_+^{M+N}$ stacks the Lagrange multipliers associated with the time-varying constraints in (11), $\frac{\delta\alpha\|\boldsymbol{\lambda}\|^2}{2}$ is the regularizing term inserted to avoid getting large values for $\boldsymbol{\lambda}$ and ensuring numerical stability of the solutions. The parameters α and δ are, respectively, constant scaling factor and step size.

Therefore, to solve the problem in (12), we use projection gradient descent to choose the primal variable $\mathbf{x}_{t+1}$ followed by a projection gradient ascent to determine the corresponding dual variable $\boldsymbol{\lambda}_{t+1}$ as functions of the decisions made in time-slot t. Leveraging the idea of the prox-method [14], we define an auxiliary function $\mathcal{A}_t(\mathbf{x}, \boldsymbol{\lambda})$, as

$$\mathcal{A}_t(\mathbf{x}, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^\top \mathbf{g}_t(\mathbf{x}) - \frac{\delta\alpha\|\boldsymbol{\lambda}\|^2}{2}. \tag{14}$$

So, at the beginning of the time-slot t, we use auxiliary values $\hat{\mathbf{x}}_t$ and $\hat{\boldsymbol{\lambda}}_t$ which are calculated at the end of previous time-slot $t-1$ to estimate the current primal and dual variables action $\mathbf{x}_t$ and $\boldsymbol{\lambda}_t$, respectively, as follows

$$\mathbf{x}_t = \mathcal{P}_{\mathcal{X}}\{\hat{\mathbf{x}}_t - \alpha\nabla_{\mathbf{x}}^\top \mathcal{A}_t(\hat{\mathbf{x}}_t, \hat{\boldsymbol{\lambda}}_t)\}, \tag{15}$$

where α is a positive stepsize, $\nabla_{\mathbf{x}}^\top \mathcal{A}_t(\hat{\mathbf{x}}_t, \hat{\boldsymbol{\lambda}}_t)$ is the gradient of $\mathcal{A}_t(\mathbf{x}, \hat{\boldsymbol{\lambda}}_t)$ with respect to $\mathbf{x}$ at point $\mathbf{x} = \hat{\mathbf{x}}_t$. Similarly,

$$\boldsymbol{\lambda}_t = \mathcal{P}_{\mathbb{R}_+^{M+N}}\left\{\hat{\boldsymbol{\lambda}}_t + \mu\nabla_{\boldsymbol{\lambda}} \mathcal{A}_t(\hat{\mathbf{x}}_t, \hat{\boldsymbol{\lambda}}_t)\right\}, \tag{16}$$

where μ is a positive stepsize. After the actions have been made, the cost function and constraint function will be revealed at the end of the time-slot. Then, we can update the auxiliary variables $\hat{\mathbf{x}}_{t+1}$, $\hat{\boldsymbol{\lambda}}_{t+1}$ based on the dual problem in (12) as

$$\hat{\mathbf{x}}_{t+1} = \mathcal{P}_{\mathcal{X}}\{\hat{\mathbf{x}}_t - \alpha\nabla_{\mathbf{x}}^\top \mathcal{L}_t^r(\mathbf{x}_t, \boldsymbol{\lambda}_t)\}, \tag{17}$$

where $\nabla_{\mathbf{x}}^\top \mathcal{L}_t^r(\mathbf{x}_t, \boldsymbol{\lambda}_t) = \nabla^\top \psi_t(\mathbf{x}_t) + \nabla^\top \mathbf{g}_t(\mathbf{x}_t)\boldsymbol{\lambda}_t = \mathbf{c}_t + \mathbf{A}^\top \boldsymbol{\lambda}_t$, and

$$\hat{\boldsymbol{\lambda}}_{t+1} = \mathcal{P}_{\mathbb{R}_+^{M+N}}\left\{\hat{\boldsymbol{\lambda}}_t + \mu\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t^r(\mathbf{x}_t, \boldsymbol{\lambda}_t)\right\}, \tag{18}$$

where $\nabla_{\boldsymbol{\lambda}} \mathcal{L}_t^r(\mathbf{x}_t, \boldsymbol{\lambda}_t) = \mathbf{g}_t(\mathbf{x}_t) - \delta\alpha\boldsymbol{\lambda}_t = \mathbf{b}_t + \mathbf{A}\mathbf{x}_t - \delta\alpha\boldsymbol{\lambda}_t$.

Algorithm 1 Mirror prox method for online optimization

1: **Initialize:** primal auxiliary iterate $\hat{\mathbf{x}}_1$ and the dual auxiliary iterate $\hat{\boldsymbol{\lambda}}_1$, the scaling factor δ and the step sizes α and μ.
2: **for** $t = 1, 2, ..., T$ **do**
3: Compute the primal variable $\mathbf{x}_t$ via (15) and dual variable $\boldsymbol{\lambda}_t$ via (16),
4: The delay cost $\psi_t(\mathbf{x}_t)$ and workload $\mathbf{g}_t(\mathbf{x}_t)$ are revealed,
5: Update the primal auxiliary variable $\hat{\mathbf{x}}_{t+1}$ via (17) and the dual auxiliary variable $\hat{\boldsymbol{\lambda}}_{t+1}$ via (18),
6: **end for**

B. Performance analysis

We first make the following two remarks and one assumption:

Remark 1. *The cost function $\psi_t(\mathbf{x})$ has a bounded gradient on $\mathcal{X}$ for each time-slot t, that is, there exists a constant $G > 0$ such that $\|\nabla\psi_t(\mathbf{x})\| \leq G, \forall \mathbf{x} \in \mathcal{X}$.*

Proof: From (5)*, we can write $\nabla\psi_t(\mathbf{x}) = \mathbf{c}_t$. This completes the proof, because $\mathbf{c}_t$ contains parameters quantifying the computation intensity of the tasks, the CPU cycle frequencies and the channel parameters (See Section III) that are all bounded values.* ■

Remark 2. *There exists a constant R such that $\|\mathbf{x}\| \leq R, \forall \mathbf{x} \in \mathcal{X}$.*

Proof: As described in Sec III, the feasible area of $\mathbf{x}$ is $\mathcal{X} = \{\mathbf{0} \leq \mathbf{x}_t \leq \mathbf{x}_u\}$, where $\mathbf{x}_u$ contains the parameters quantifying the computing capability of the SDs and the servers as well as the transmission capacity that are all bounded values. ■

Assumption 1. *There exists a constant $\zeta \geq 0$ and an interior point $\tilde{\mathbf{x}}_t \in \mathcal{X}$ such that $\mathbf{g}_t(\tilde{\mathbf{x}}_t) \leq -\zeta\mathbf{1}, \forall t$.*

The Assumption 1 is Slater's condition, which guarantees zero duality gap [15]. Under the Slater condition, the primal problem in (11) and its dual problem have the same optimal objective value, and a dual optimal solution $\boldsymbol{\lambda}_t^*$ exists and is finite, i.e., $\boldsymbol{\lambda}_t^* \leq \infty$. The physical interpretation of the assumption of Slater's condition holding true in our developments is that the amount of incoming workloads should be proportional to the available resources in terms of the overall computational capabilities of the edge servers. For instance, if a large number of users with an excessive amount of workload per user are to be scheduled at the same time to be served continuously by only one server with insufficient computational resources, the queue stability of the network, i.e., the strict feasibility of the constraints in (11), cannot be assured, even in the long run. Therefore, Assumption 1 assures that there always exists an optimal and a feasible solution at each time-slot, which is only obtainable by a Genie who has the knowledge of future, so that the per time-slot constraints in (11) are satisfied. In fact, the Assumption 1 assures that there always exists a dynamic optimum solution $\mathbf{x}_t^*$ in (10) that evaluates the performance of the proposed algorithm by computing the incurred regret.

Then, we introduce the following three Lemmas:

Lemma 1. *For an non-square $(M + N) \times (M + MN + N)$ incidence matrix $\mathbf{A}$ shown in* (7)*, we have $\|\mathbf{Ax}\|^2 \leq (M + N + 2MN)\|\mathbf{x}\|^2$.*

Proof: See Appendix B. ■

Lemma 2. *For any time-slot t, the auxiliary function $\mathcal{A}_t(\mathbf{x}, \boldsymbol{\lambda})$ has Lipschitz continuous gradient, expressed as*

$$\begin{aligned}&\left\|\nabla_{\mathbf{x}}\mathcal{A}_t(\mathbf{x}, \boldsymbol{\lambda}) - \nabla_{\mathbf{x}'}\mathcal{A}_t(\mathbf{x}', \boldsymbol{\lambda}')\right\|^2 + \\ &\left\|\nabla_{\boldsymbol{\lambda}}\mathcal{A}_t(\mathbf{x}, \boldsymbol{\lambda}) - \nabla_{\boldsymbol{\lambda}'}\mathcal{A}_t(\mathbf{x}', \boldsymbol{\lambda}')\right\|^2 \\ \leq& 2(M + N + 2MN + \delta^2\alpha^2)\left(\left\|\mathbf{x} - \mathbf{x}'\right\|^2 + \left\|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\right\|^2\right). \quad (19)\end{aligned}$$

Proof: See Appendix C. ■

Lemma 3. *For any time-slot t, $\boldsymbol{\lambda}_t^* \in \mathbb{R}_+^{M+N}$ and $\mathbf{x}_t^* \in \arg\min_{\mathbf{x}\in\mathcal{X}} \psi_t(\mathbf{x})$ s. t $\mathbf{g}_t(\mathbf{x}) \leq \mathbf{0}$, setting $\max(\alpha, \mu)$ $(M + N + 2MN + \delta^2\alpha^2) \leq \frac{1}{4}$, we have*

$$\begin{aligned}&\mathcal{L}_t^r(\mathbf{x}_t, \boldsymbol{\lambda}_t^*) - \mathcal{L}_t^r(\mathbf{x}_t^*, \boldsymbol{\lambda}_t) \leq \\ &\frac{\left\|\mathbf{x}_t^* - \hat{\mathbf{x}}_t\right\|^2 - \left\|\mathbf{x}_t^* - \hat{\mathbf{x}}_{t+1}\right\|^2}{2\max(\alpha, \mu)} + \frac{\left\|\boldsymbol{\lambda}_t^* - \hat{\boldsymbol{\lambda}}_t\right\|^2 - \left\|\boldsymbol{\lambda}_t^* - \hat{\boldsymbol{\lambda}}_{t+1}\right\|^2}{2\max(\alpha, \mu)} \\ &+ \max(\alpha, \mu)\|\nabla\psi_t(\mathbf{x}_t)\|^2. \quad (20)\end{aligned}$$

Proof: See Appendix D. ■

Theorem 1. *The dynamic regret is bounded by*

$$-RGT \leq R_T \leq \frac{R^2 + R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha, \mu)} + \alpha TG^2, \quad (21)$$

where $\mathcal{V}_{\mathbf{x}_t^*}^T = \sum_{t=2}^T \left(\left\|\mathbf{x}_{t-1}^* - \mathbf{x}_t^*\right\|\right)$ is the accumulated hindsight optimum variations, between the consecutive time-slots.

Proof: See Appendix E. ■

Theorem 2. *The aggregate violation is upper-bounded by*

$$V_T \leq \left[\left(\frac{2R^2}{\max(\alpha, \mu)} + \frac{2R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha, \mu)} + 2\alpha TG^2 + 2RGT\right)\left(\delta\alpha T + \frac{1}{\max(\alpha, \mu)}\right)\right]^{1/2}. \quad (22)$$

Proof: See Appendix F. ■

According to Theorems 1 and 2, the upper bounds of the dynamic regret and the aggregate violation depend on some constants defined in the remarks, the pre-selected scaling factor, the primal and dual step sizes, the time horizon and the accumulated hindsight optimum variation.

Corollary 1. *Letting the primal and the dual step sizes be equal according $\alpha = \mu = O(T^{-\frac{1}{3}})$, the scaling factor be $\delta = T^{-\frac{2}{3}}$, one can express the upper bound of the dynamic regret in* (21) *as*

$$R_T \leq O\left(\max\left\{T^{\frac{1}{3}}\mathcal{V}_{\mathbf{x}_t^*}^T, T^{\frac{2}{3}}\right\}\right), \quad (23)$$

and the upper bound of the aggregate violation in (22) *as*

$$V_T \leq O\left(\max\left\{T^{\frac{1}{3}}\sqrt{\mathcal{V}_{\mathbf{x}_t^*}^T}, T^{\frac{2}{3}}\right\}\right). \quad (24)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2020.2973566, IEEE Transactions on Communications

Corollary 1 states that if the optimal hindsight variation is sub-linear according to $\mathcal{V}_{\mathbf{x}_t^*}^T = o(T)$, then the aggregate violation V_T is sub-linear over the time horizon T. Furthermore, if the optimal hindsight variation is sub-linear according to $\mathcal{V}_{\mathbf{x}_t^*}^T = o(T^{\frac{2}{3}})$, then both of the dynamic regret R_T and the aggregate violation V_T achieve sub-linear upper-bounds over the time horizon T.

C. Distributed Mirror Prox (DMP) for Online Optimization

In addition to the closed-form updates, DMP can also perform a fully decentralized implementation by exploiting the network structure. We use the cost functions for each SD locally processing $f_t^m(d_t^m)$, offloading $f_{tr,t}^{mn}(a_t^{mn})$ and each MEC server processing $f_{ex,t}^n(z_t^n)$ which are defined in Section II to perform each individual SD's and each individual MEC server's primal-dual auxiliary updates. Per time-slot t, the primal update of each SD's actions and each MEC server's action can be expressed as

$$d_t^m = \left[\hat{d}_t^m + \alpha\hat{\lambda}_t^m\right]_0^{d_u}, \tag{25}$$

where $[\,\cdot\,]_0^{d_u} = \min\{d_u, \max\{\cdot\,, 0\}\}$ and α is the step size.

$$a_t^{mn} = \left[\hat{a}_t^{mn} + \alpha(\hat{\lambda}_t^m - \hat{\lambda}_t^n)\right]_0^{a_u}, \forall n \in \mathcal{N}, \tag{26}$$

$$z_t^n = \left[\hat{z}_t^n + \alpha\hat{\lambda}_t^n\right]_0^{z_u}, \tag{27}$$

and the dual updates of each SD and each MEC server reduce to

$$\lambda_t^m = \left[\hat{\lambda}_t^m + \mu(b_t^m - \hat{d}_t^m - \sum_{n\in\mathcal{N}} \hat{a}_t^{mn})\right]^+, \tag{28}$$

where $[\,\cdot\,]^+ = \max\{\cdot\,, 0\}$ and μ is the step size.

$$\lambda_t^n = \left[\hat{\lambda}_t^n + \mu(\sum_{m\in\mathcal{M}} \hat{a}_t^{mn} - \hat{z}_t^n)\right]^+. \tag{29}$$

Then, the primal auxiliary updates can be given by

$$\hat{d}_{t+1}^m = \left[\hat{d}_t^m - \alpha(\nabla f_t^m(d_t^m) - \lambda_t^m)\right]_0^{d_u}, \tag{30}$$

$$\hat{a}_{t+1}^{mn} = \left[\hat{a}_t^{mn} - \alpha\nabla f_{tr,t}^{mn}(a_t^{mn}) + \alpha(\lambda_t^m - \lambda_t^n)\right]_0^{a_u}, \forall n \in \mathcal{N}, \tag{31}$$

$$\hat{z}_{t+1}^n = \left[\hat{z}_t^n - \alpha(\nabla f_{ex,t}^n(z_t^n) - \lambda_t^n)\right]_0^{z_u}, \tag{32}$$

and the dual auxiliary updates are

$$\hat{\lambda}_{t+1}^m = \left[\hat{\lambda}_t^m + \mu(b_t^m - d_t^m - \sum_{n\in\mathcal{N}} a_t^{mn} - \delta\alpha\lambda_t^m)\right]^+, \tag{33}$$

$$\hat{\lambda}_{t+1}^n = \left[\hat{\lambda}_t^n + \mu(\sum_{m\in\mathcal{M}} a_t^{mn} - z_t^n - \delta\alpha\lambda_t^n)\right]^+. \tag{34}$$

The computational steps of the proposed DMP algorithm is shown in Fig.2. Based on the locally available information, each SD makes a decision at the start of a current time-slot t on how much to process locally and how much to offload by calculating d_t^m and a_t^{mn} as well as calculating the Lagrange multiplier $\boldsymbol{\lambda}_t^m$. While the SD is calculating, the MEC LC calculates in parallel the amount of workload processed by the servers using (27) and the Lagrange multiplier $\boldsymbol{\lambda}_t^n$. At the end of the time-slot t, the MEC broadcasts its state parameters $\boldsymbol{\lambda}_t^n$ and the state information of the wireless links r_t^m as common information to all SDs. Finally, each SD autonomously updates its auxiliary variables $\hat{d}_{t+1}^m$, $\hat{a}_{t+1}^{mn}$, $\hat{z}_{t+1}^n$ and $\hat{\lambda}_{t+1}^m$ based on current status (e.g CPU frequency ρ_t^m), which will be used to make next time-slot $t+1$ decisions.

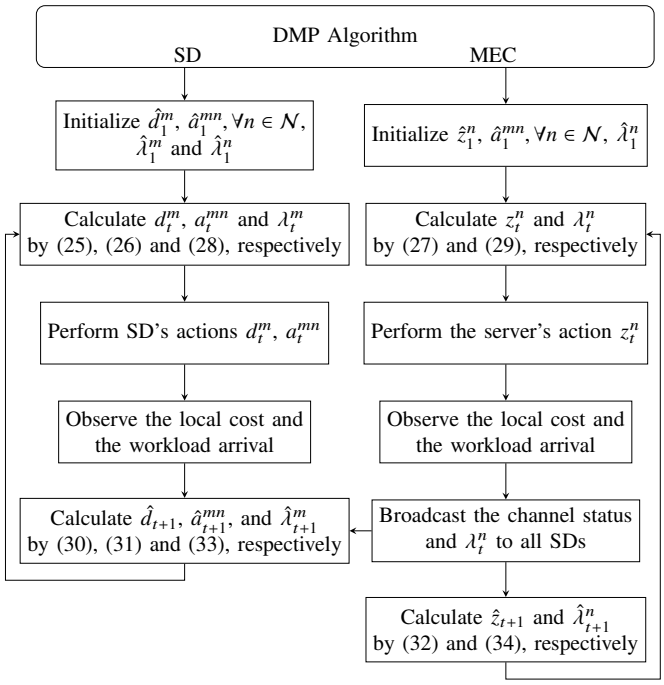


Fig. 2. DMP algorithm flowchart

Remark 3. *DMP Algorithm is distributed in the sense that there is no exchange of information amongst SDs to ensure queue stability at the servers. Furthermore, the cooperation between SDs and MEC servers is based on a single broadcast of channel information and MEC servers' states. The auxiliary variables better assist SDs and MEC servers to make more accurate decisions to minimize the overall network delay and ensure all queues stability.*

V. Simulation Results

We assume M smart devices are located at an equal distance of 150m from N MEC servers. The wireless channel bandwidth is W = 10MHz. The transmission power $p_{tr,t}^m$ is ranged from 50mW to 150mW with uniform random distribution. The noise power spectral density is $N_0 = -174$dBm/Hz. The small-scale fading channel power gains are exponentially distributed with unit mean, i.e., $\gamma_t^m \sim Exp(1), m \in \mathcal{M}$. We consider the channel gain is $h_t^m = \gamma_t^m q_0 (l_0/l_t^m)^\theta$, where l_0 = 1m is the reference distance, l_t^m = 150m is the distance between the mth SD and the BS, $q_0 = -40$dB is the path-loss constant, and $\theta = 4$ is the path-loss exponent [16]. During each time-slot of τ = 1ms, the arriving task b_t^m is uniformly and independently distributed within $[0, 8]$ kbits. The computation intensities w_t^m is randomly and independently ranged from 500cycles/bit to 5000cycles/bit with uniform distribution. The

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2020.2973566, IEEE Transactions on Communications

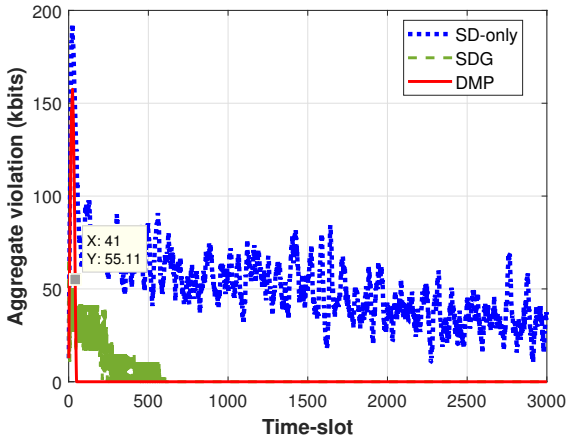


Fig. 3. Aggregate Violation V_T for SD-only, SDG and DMP (M=10, N=5).

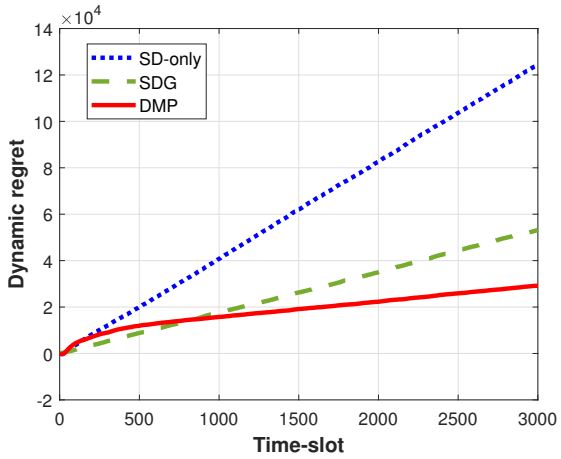


Fig. 4. Dynamic regret R_T for SD-only, SDG and DMP (M=10, N=5).

CPU-cycle frequency of SDs (ρ_t^m) and the MEC servers (ρ_t^n) are randomly and independently chosen from 1GHz to 2.5GHz (cycles/sec) and 2.5GHz to 5GHz (cycles/sec), respectively, with uniform distributions [17]. Then, the upper limits of the computing capabilities of the SDs and the MEC servers as well as the transmission capacity per time-slot are d_u = 5kbits, z_u = 10kbits and a_u = 3kbits, respectively.

In simulations, the proposed DMP is benchmarked by the dynamic optimum, the stochastic dual gradient (SDG) [18] and the SD-only, i.e., without computation offloading. Figs. 3, 4, 5 and 7 are under the 10 SDs and 5 MEC servers, i.e., M=10 and N=5, case. In Fig. 3, we show the aggregation violations of DMP, SD-only and SDG fall in different time-slots. Although the DMP scheme has higher violations than the SDG scheme before 41 time-slots, the DMP gets faster convergence than the SDG as well as SD-only and performs better by approaching very close values to zero over the long time horizon. Due to the limited computational resource of the SDs, the SD-only scheme cannot get convergence for the aggregation violation even at the end of time horizon. Furthermore, the DMP achieves lower computation complexity than the SDG which requires longer time to deal with a convex

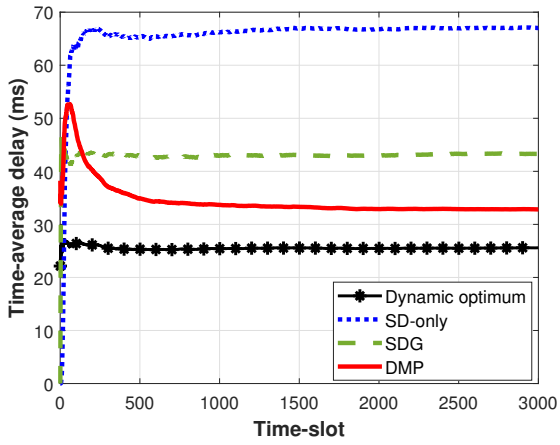


Fig. 5. Time-average network delay for dynamic optimum, SD-only, SDG and DMP (M=10, N=5).

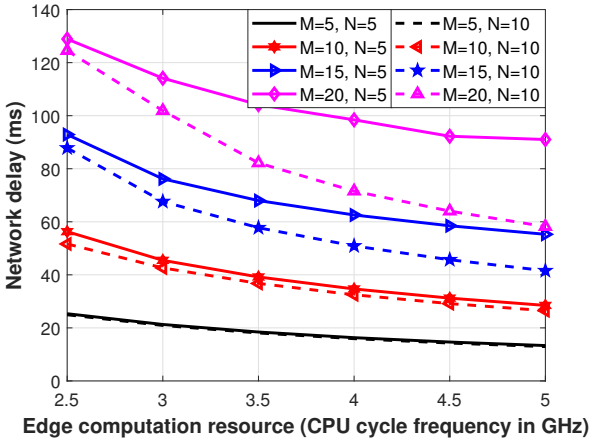


Fig. 6. The impact of the network delay over different CPU cycle frequency of MEC servers and different number of SDs via the DMP scheme.

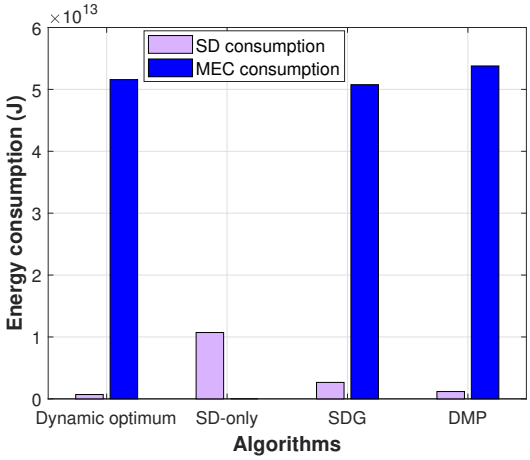


Fig. 7. SDs and MEC servers energy consumption over time-horizon T for dynamic optimum, SD-only, SDG and DMP (M=10, N=5).

program.

Then, we compare dynamic regret for DMP, SDG and SD-only in Fig. 4. It can be observed that the dynamic regret of DMP is linear over a very short period of time in the beginning, because the proposed algorithm has just started and has not yet adapted to the environment. But soon after, the growing rate of the dynamic regret of DMP is slower than that of SD-only and SDG, which verifies the results in Fig. 5. Fig. 5 shows a steady-state improvement of about 39% when both DMP and SDG are compared against the dynamic optimum. Notice that the dynamic optimum is the lower bound in an ideal world, when all future information are assumed to be available at the time of decision making. Although, this steady-state improvement in delay comes at the price of an overshoot over a short transient time in DMP, it also pays off in terms of aggregate violation performance, as shown in Fig. 3, when compared with the SDG.

In Fig. 6, we use the network delay that has converged, i.e., the network delay at the end of the time horizon, to investigate the impact by different levels of MEC servers' CPU cycle frequency and different number of SDs. As Fig. 6 demonstrates, the network delay decreases with the promotion of the MEC servers' CPU cycle frequency. Furthermore, with the number of SDs increasing, the network delay increases due to more computation-intensive tasks that required to be executed by MEC servers. For M=5 and M=10, i.e., 5 SDs and 10 SDs, increasing the CPU cycle frequency and the number of MEC servers does not have great effect on the network delay due to the smaller network load compared to M=15 and M=20 cases. For M=20 case, i.e., 20 SDs, the network delay does not have great difference by increasing the CPU cycle frequency from 4.5GHz to 5GHz. However, the network delay has a considerable reduction by increasing the number of MEC servers from N=5 to N=10. This is because the limited buffers of 5 MEC servers, i.e., N=5, are hard to support the larger network load. The proposed algorithm not only aims to minimize the delay but ensure the queue stability. Thus, once the edge has enough computational resources and buffer resources, e.g., N=10 case, the network delay can be effectively reduced.

The aggregate energy consumption in Fig. 7 and Table. II. reveal that, although, the DMP requires higher total energy consumption or MEC servers' energy consumption than the other schemes, i.e., SDG, SD-only and dynamic optimum, its SDs' energy consumption is much lower than the other schemes. Specifically, the MEC servers' energy consumption of the DMP exceeds that of the SDG by around 6%, but the SDs' energy consumption of the DMP has a about 56% reduction compared to that of the SDG. Because the CPU cycle frequency of the MEC server is much higher than that of SD, and the equations (3) and (4) are quadratic with respect to the CPU cycle frequency, the MEC server consumes more energy than SD under the same amount of tasks. This result implies that the DMP is much more efficient than the other schemes in allocating computation-intensive tasks to the MEC and enduring the battery life of the SDs as a result.

TABLE II
THE ENERGY CONSUMPTION STATISTICS FOR DYNAMIC OPTIMUM, SD-ONLY, SDG AND DMP.

	MEC servers	SDs	Total
Optimum	$5.16 * 10^{13}$ (J)	$6.70 * 10^{11}$ (J)	$5.23 * 10^{13}$ (J)
SD-only	0 (J)	$1.07 * 10^{13}$ (J)	$1.07 * 10^{13}$ (J)
SDG	$5.08 * 10^{13}$ (J)	$2.66 * 10^{12}$ (J)	$5.35 * 10^{13}$ (J)
DMP	$5.38 * 10^{13}$ (J)	$1.18 * 10^{12}$ (J)	$5.50 * 10^{13}$ (J)

VI. CONCLUSION

An online optimization problem was formulated to minimize the overall long-term average of the network delay for computation offloading in the MEC system. We developed a new DMP algorithm and found upper-bounds for the resulting dynamic regret and the aggregate violation. And these upper-bounds can be sub-linear under sub-linear accumulated hindsight optimum variations. The simulation results confirm that the DMP algorithm can achieve lower overall network delay than the other schemes, considered for comparison. Although, this improvement comes at the price of higher joint energy consumption of smart devices and the MEC servers, the energy consumption of smart devices is lower than the other schemes. This is an interesting result from the perspective of smart devices in the sense that the SDs are more strained and vulnerable to the shortage of energy, which is supplied by batteries, than the servers, which can be supplied by the power grid and renewable sources directly.

As a future research direction, it would be interesting to amend the optimization problem introduced in this paper with constraints ensuring an instantaneous quality of experience, i.e., a minimum level of service per time-slot, for users.

APPENDIX

A. Example

The incidence matrix $\mathbf{A}$ for a network with 3 smart devices, i.e., M=3, and 2 servers, i.e., N=2, can be constructed as

$$\mathbf{A} = \begin{bmatrix} -1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & -1 \end{bmatrix}. \quad (35)$$

B. Proof for Lemma 1

Using the characteristic of the norm-squared, we have

$$\|\mathbf{Ax}\|^2 = [\mathbf{Ax}]^\top \mathbf{Ax} = \mathbf{x}^\top \mathbf{A}^\top \mathbf{Ax}. \quad (36)$$

The incidence matrix $\mathbf{A}$ can be denoted as $\mathbf{A} = \mathbf{U}[\Sigma \vdots \mathbf{0}]\mathbf{V}^\top$, where $\mathbf{U}$ and $\mathbf{V}$ are $(M+N)\times(M+N)$ and $(M+MN+N)\times(M+MN+N)$ unitary matrices, respectively, Σ is a $(M+N)\times(M+N)$ diagonal matrix with singular values $\sigma(\mathbf{A})$ lying on the diagonal, $\mathbf{0}$ is $(M+N)\times MN$ matrix with all zeros entries. Then, we have

$$\mathbf{A}^\top \mathbf{A} = \mathbf{V} \begin{bmatrix} \Sigma^2 & \vdots & \mathbf{0} \\ \cdots & \vdots & \cdots \\ \mathbf{0}^\top & & \mathbf{0} \end{bmatrix} \mathbf{V}^\top. \quad (37)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2020.2973566, IEEE Transactions on Communications

Combining (37) and (36), we have

$$\|\mathbf{Ax}\|^2 = \mathbf{x}^\top \mathbf{V} \begin{bmatrix} \Sigma^2 & \vdots & \mathbf{0} \\ \cdots & \vdots & \cdots \\ \mathbf{0}^\top & & \mathbf{0} \end{bmatrix} \mathbf{V}^\top \mathbf{x}. \tag{38}$$

Then, we denote $\mathbf{V}^\top\mathbf{x} = [\mathbf{s}^\top \vdots \mathbf{h}^\top]^\top$ and $\mathbf{x}^\top\mathbf{V} = [\mathbf{s}^\top \vdots \mathbf{h}^\top]$, where $\mathbf{s}$ and $\mathbf{h}$ are column vectors with $(M+N)$ and (MN) rows, respectively. Thus, we can update (38), given by

$$\begin{aligned} \|\mathbf{Ax}\|^2 &= \mathbf{s}^\top\Sigma^2\mathbf{s} \\ &\overset{(a)}{\leq} \sigma^2_{\max}(\mathbf{A})\|\mathbf{s}\|^2 \\ &\overset{(b)}{=} \lambda_{\max}(\mathbf{AA}^\top)\|\mathbf{s}\|^2 \\ &\overset{(c)}{\leq} Tr(\mathbf{AA}^\top)\|\mathbf{s}\|^2 \\ &\overset{(d)}{\leq} Tr(\mathbf{AA}^\top)\|\mathbf{x}\|^2, \end{aligned} \tag{39}$$

where (a) chooses the maximum singular value $\sigma_{\max}$ from the diagonal matrix Σ, (b) is due to $\mathbf{AA}^\top = \mathbf{U}\Sigma^2\mathbf{U}$, (c) is due to the sum of the eigenvalues of $\mathbf{AA}^\top$ is the same as the trace of $\mathbf{AA}^\top$, (d) is due to $\|\mathbf{s}\|^2 + \|\mathbf{h}\|^2 = \mathbf{x}^\top\mathbf{VV}^\top\mathbf{x} = \|\mathbf{x}\|^2$. Based on the characteristic of the incidence matrix $\mathbf{A}$ shown in (7), we have

$$\mathbf{AA}^\top = \begin{bmatrix} \mathbf{I}_{M\times M} + N\times\mathbf{1}_{M\times M} & \vdots & -\mathbf{1}_{M\times N} \\ \cdots\cdots\cdots & & \cdots\cdots\cdots \\ -\mathbf{1}_{N\times M} & \vdots & M\times\mathbf{1}_{N\times N} + \mathbf{I}_{N\times N} \end{bmatrix}, \tag{40}$$

where $\mathbf{1}_{i\times j}$ is defined as a $i\times j$ matrix with all one elements. So, by calculating the trace of $\mathbf{AA}^\top$ which is $(M+N+2MN)$, we complete the proof.

C. Proof for Lemma 2

We use the auxiliary function in (14) to expand the following two parts, given by

$$\begin{aligned} &\left\|\nabla_{\mathbf{x}}\mathcal{A}_t(\mathbf{x},\boldsymbol{\lambda}) - \nabla_{\mathbf{x}'}\mathcal{A}_t(\mathbf{x}',\boldsymbol{\lambda}')\right\|^2 \\ =&\left\|\boldsymbol{\lambda}^\intercal\nabla\mathbf{g}_t(\mathbf{x}) - \boldsymbol{\lambda}'^\intercal\nabla\mathbf{g}_t(\mathbf{x}')\right\|^2 \\ =&\left\|\mathbf{A}^\top(\boldsymbol{\lambda}-\boldsymbol{\lambda}')\right\|^2 \overset{(a)}{\leq} (M+N+2MN)\left\|\boldsymbol{\lambda}-\boldsymbol{\lambda}'\right\|^2, \end{aligned} \tag{41}$$

where (a) uses the Lemma 1.

$$\begin{aligned} &\left\|\nabla_{\boldsymbol{\lambda}}\mathcal{A}_t(\mathbf{x},\boldsymbol{\lambda}) - \nabla_{\boldsymbol{\lambda}'}\mathcal{A}_t(\mathbf{x}',\boldsymbol{\lambda}')\right\|^2 \\ =&\left\|\mathbf{g}_t(\mathbf{x}) - \delta\alpha\boldsymbol{\lambda} - \mathbf{g}_t(\mathbf{x}') + \delta\alpha\boldsymbol{\lambda}'\right\|^2 \\ =&\left\|\mathbf{A}(\mathbf{x}-\mathbf{x}') - \delta\alpha(\boldsymbol{\lambda}-\boldsymbol{\lambda}')\right\|^2 \\ \overset{(b)}{\leq}& 2\left\|\mathbf{A}(\mathbf{x}-\mathbf{x}')\right\|^2 + 2\delta^2\alpha^2\left\|\boldsymbol{\lambda}-\boldsymbol{\lambda}'\right\|^2 \\ \overset{(c)}{\leq}& 2(M+N+2MN)\left\|\mathbf{x}-\mathbf{x}'\right\|^2 + 2\delta^2\alpha^2\left\|\boldsymbol{\lambda}-\boldsymbol{\lambda}'\right\|^2, \end{aligned} \tag{42}$$

where (b) uses $(a+b)^2 \leq 2(a^2+b^2)$, (c) also uses the Lemma 1. Adding (41) and (42), we have

$$\begin{aligned} &(41)+(42) \\ \leq& 2(M+N+2MN)\left\|\mathbf{x}-\mathbf{x}'\right\|^2 + (M+N+2MN \\ &\qquad + 2\delta^2\alpha^2)\left\|\boldsymbol{\lambda}-\boldsymbol{\lambda}'\right\|^2 \\ \leq& 2(M+N+2MN+\delta^2\alpha^2)\left(\left\|\mathbf{x}-\mathbf{x}'\right\|^2 + \left\|\boldsymbol{\lambda}-\boldsymbol{\lambda}'\right\|^2\right). \end{aligned} \tag{43}$$

D. Proof for Lemma 3

The regularized Lagrangian function $\mathcal{L}_t^r(\cdot,\cdot)$ defined in (12) is convex with respect to the first argument and concave with respect to the second argument. Hence, using [19] and [15], we can write

$$\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t) - \mathcal{L}_t^r(\mathbf{x}_t^*,\boldsymbol{\lambda}_t) \leq \nabla_{\mathbf{x}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t)(\mathbf{x}_t-\mathbf{x}_t^*), \tag{44}$$

$$\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t^*) - \mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t) \leq (\boldsymbol{\lambda}_t^*-\boldsymbol{\lambda}_t)^\top\nabla_{\boldsymbol{\lambda}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t). \tag{45}$$

Then, adding these two equalities (44) and (45), we have

$$\begin{aligned} &\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t^*) - \mathcal{L}_t^r(\mathbf{x}_t^*,\boldsymbol{\lambda}_t) \leq \\ &\nabla_{\mathbf{x}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t)(\mathbf{x}_t-\mathbf{x}_t^*) + (\boldsymbol{\lambda}_t^*-\boldsymbol{\lambda}_t)^\top\nabla_{\boldsymbol{\lambda}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t). \end{aligned} \tag{46}$$

Let us consider $\phi(\mathbf{e}) = \frac{1}{2}\mathbf{e}^\top\mathbf{e} = \frac{1}{2}\|\mathbf{e}\|^2$, which is a differentiable and strictly convex function of vector $\mathbf{e}$. Then, according to [20], the Bregman divergence between two vectors $\mathbf{m}$ and $\mathbf{n}$ is given by $B(\mathbf{m},\mathbf{n}) = \phi(\mathbf{m}) - \phi(\mathbf{n}) - (\mathbf{m}-\mathbf{n})^\top\nabla\phi(\mathbf{n}) = \frac{1}{2}\|\mathbf{m}-\mathbf{n}\|^2$. Let us stack the corresponding primal and dual variables $(\mathbf{x}_t,\boldsymbol{\lambda}_t)$ and $(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t)$ in variables $\mathbf{m}$ and $\mathbf{n}$, respectively, as $\mathbf{m} = [\mathbf{x}_t^\top \vdots \boldsymbol{\lambda}_t^\top]^\top \in \mathbb{R}^{2M+2N+MN}$ and $\mathbf{n} = [\hat{\mathbf{x}}_t^\top \vdots \hat{\boldsymbol{\lambda}}_t^\top]^\top \in \mathbb{R}^{2M+2N+MN}$. Then, it can be easily verified that

$$B(\mathbf{m},\mathbf{n}) = \frac{\|\mathbf{x}_t-\hat{\mathbf{x}}_t\|^2}{2} + \frac{\left\|\boldsymbol{\lambda}_t-\hat{\boldsymbol{\lambda}}_t\right\|^2}{2}. \tag{47}$$

Given $\mathbf{u},\mathbf{a}$ and $\mathbf{o}$, we set

$$\begin{aligned} \mathbf{w} &= \arg\min_{\mathbf{x}\in\mathcal{X}} \mathbf{a}^\top(\mathbf{x}-\mathbf{u}) + B(\mathbf{x},\mathbf{u}), \\ \mathbf{u}_+ &= \arg\min_{\mathbf{x}\in\mathcal{X}} \mathbf{o}^\top(\mathbf{x}-\mathbf{u}) + B(\mathbf{x},\mathbf{u}). \end{aligned} \tag{48}$$

Based on analysis of [14, Lemma 3.1] and $\epsilon \geq 0$, we have

$$\begin{aligned} \epsilon\mathbf{o}^\top(\mathbf{w}-\mathbf{x}) \leq B(\mathbf{x},\mathbf{u}) - B(\mathbf{x},\mathbf{u}_+) + \frac{\epsilon^2}{\eta}\|\mathbf{a}-\mathbf{o}\|^2 \\ - \frac{\eta}{2}\left[\|\mathbf{w}-\mathbf{u}\|^2 + \|\mathbf{w}-\mathbf{u}_+\|^2\right]. \end{aligned} \tag{49}$$

To leverage (49) and (47), we define $\mathbf{o},\mathbf{w},\mathbf{x},\mathbf{u},\mathbf{u}_+$, $\mathbf{a}$, ϵ and η as follows $\mathbf{x} = \left[\mathbf{x}_t^{*\top} \vdots \boldsymbol{\lambda}_t^{*\top}\right]^\top$, $\mathbf{w} = \left[\mathbf{x}_t^\top \vdots \boldsymbol{\lambda}_t^\top\right]^\top$, $\mathbf{u} = \left[\hat{\mathbf{x}}_t^\top \vdots \hat{\boldsymbol{\lambda}}_t^\top\right]^\top$, $\mathbf{u}_+ = \left[\hat{\mathbf{x}}_{t+1}^\top \vdots \hat{\boldsymbol{\lambda}}_{t+1}^\top\right]^\top$, $\mathbf{a} = \left[\nabla_{\hat{\mathbf{x}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t) \;\vdots\; -\nabla_{\hat{\boldsymbol{\lambda}}}^\top\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t)\right]^\top$, $\mathbf{o} = \left[\nabla_{\mathbf{x}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t) \;\vdots\; -\nabla_{\boldsymbol{\lambda}}^\top\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t)\right]^\top$, $\epsilon = \max(\alpha,\mu)$ and $\eta = 1$. Putting them into (49) and using (46), we can easily get

$$\begin{aligned} &\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t^*) - \mathcal{L}_t^r(\mathbf{x}_t^*,\boldsymbol{\lambda}_t) \leq \\ &\frac{\left\|\mathbf{x}_t^*-\hat{\mathbf{x}}_t\right\|^2 - \left\|\mathbf{x}_t^*-\hat{\mathbf{x}}_{t+1}\right\|^2}{2\max(\alpha,\mu)} + \frac{\left\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_t\right\|^2 - \left\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_{t+1}\right\|^2}{2\max(\alpha,\mu)} \\ &+ \underbrace{\frac{\max(\alpha,\mu)}{2}\left(\left\|\nabla_{\hat{\mathbf{x}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t) - \nabla_{\mathbf{x}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t)\right\|^2\right)}_{\mathbf{I}} \\ &+ \underbrace{\frac{\max(\alpha,\mu)}{2}\left(\left\|\nabla_{\hat{\boldsymbol{\lambda}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t) - \nabla_{\boldsymbol{\lambda}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t)\right\|^2\right)}_{\mathbf{II}} \\ &- \frac{1}{2}\left(\|\hat{\mathbf{x}}_t-\mathbf{x}_t\|^2 + \left\|\hat{\boldsymbol{\lambda}}_t-\boldsymbol{\lambda}_t\right\|^2\right). \end{aligned} \tag{50}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2020.2973566, IEEE Transactions on Communications

Then, we upper bound $(\mathbf{I}+\mathbf{II})$, given by

$$\begin{aligned}
&\mathbf{I}+\mathbf{II}\\
&\overset{(a)}{\leq} \frac{\max(\alpha,\mu)}{2}\Big(2\Big\|\nabla_{\hat{\mathbf{x}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t)-\nabla_{\mathbf{x}}\mathcal{A}_t(\mathbf{x}_t,\boldsymbol{\lambda}_t)\Big\|^2\\
&\quad+2\|\nabla\psi_t(\mathbf{x}_t)\|^2+\Big\|\nabla_{\hat{\boldsymbol{\lambda}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t)-\nabla_{\boldsymbol{\lambda}}\mathcal{A}_t(\mathbf{x}_t,\boldsymbol{\lambda}_t)\Big\|^2\Big)\\
&\leq \max(\alpha,\mu)\Big(\Big\|\nabla_{\hat{\mathbf{x}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t)-\nabla_{\mathbf{x}}\mathcal{A}_t(\mathbf{x}_t,\boldsymbol{\lambda}_t)\Big\|^2\\
&\quad+\|\nabla\psi_t(\mathbf{x}_t)\|^2+\Big\|\nabla_{\hat{\boldsymbol{\lambda}}}\mathcal{A}_t(\hat{\mathbf{x}}_t,\hat{\boldsymbol{\lambda}}_t)-\nabla_{\boldsymbol{\lambda}}\mathcal{A}_t(\mathbf{x}_t,\boldsymbol{\lambda}_t)\Big\|^2\Big)\\
&\overset{(b)}{\leq} \max(\alpha,\mu)\|\nabla\psi_t(\mathbf{x}_t)\|^2+\max(\alpha,\mu)2(M+N+2MN\\
&\qquad+\delta^2\alpha^2)\Big(\|\hat{\mathbf{x}}_t-\mathbf{x}_t\|^2+\Big\|\hat{\boldsymbol{\lambda}}_t-\boldsymbol{\lambda}_t\Big\|^2\Big),
\end{aligned}\tag{51}$$

where (a) first expanding the gradient $\nabla_{\mathbf{x}}\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t)$, and then using the inequality $(a+b)^2\leq 2(a^2+b^2)$; (b) using Lemma 2. Then, combine (50) and (51), we have

$$\begin{aligned}
\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t^*)-\mathcal{L}_t^r(\mathbf{x}_t^*,\boldsymbol{\lambda}_t)&\leq\frac{\Big\|\mathbf{x}_t^*-\hat{\mathbf{x}}_t\Big\|^2-\Big\|\mathbf{x}_t^*-\hat{\mathbf{x}}_{t+1}\Big\|^2}{2\max(\alpha,\mu)}\\
&+\frac{\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_t\Big\|^2-\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_{t+1}\Big\|^2}{2\max(\alpha,\mu)}+\max(\alpha,\mu)\|\nabla\psi_t(\mathbf{x}_t)\|^2\\
&+\Big[\max(\alpha,\mu)2(M+N+2MN+\delta^2\alpha^2)\\
&\qquad-\frac{1}{2}\Big]\Big(\|\hat{\mathbf{x}}_t-\mathbf{x}_t\|^2+\Big\|\hat{\boldsymbol{\lambda}}_t-\boldsymbol{\lambda}_t\Big\|^2\Big).
\end{aligned}\tag{52}$$

By setting $\max(\alpha,\mu)(M+N+2MN+\delta^2\alpha^2)\leq\frac{1}{4}$, we complete the proof.

E. Proof for Theorem 1

By summing up Lemma 3 (20) over the entire time horizon $t=1,...,T$, we first upper bound $\frac{1}{2\max(\alpha,\mu)}\sum_{t=1}^T\Big(\Big\|\mathbf{x}_t^*-\hat{\mathbf{x}}_t\Big\|^2-\Big\|\mathbf{x}_t^*-\hat{\mathbf{x}}_{t+1}\Big\|^2\Big)$ and $\frac{1}{2\max(\alpha,\mu)}\sum_{t=1}^T\Big(\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_t\Big\|^2-\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_{t+1}\Big\|^2\Big)$, respectively, as

$$\begin{aligned}
&\frac{1}{2\max(\alpha,\mu)}\sum_{t=1}^T\Big(\Big\|\mathbf{x}_t^*-\hat{\mathbf{x}}_t\Big\|^2-\Big\|\mathbf{x}_t^*-\hat{\mathbf{x}}_{t+1}\Big\|^2\Big)\\
&=\frac{1}{2\max(\alpha,\mu)}\sum_{t=1}^T\Big(\|\hat{\mathbf{x}}_t\|^2-\|\hat{\mathbf{x}}_{t+1}\|^2\Big)+\\
&\quad\frac{1}{2\max(\alpha,\mu)}\sum_{t=1}^T2\Big\|\mathbf{x}_t^*\Big\|\Big(\|\hat{\mathbf{x}}_{t+1}\|-\|\hat{\mathbf{x}}_t\|\Big)\\
&=\frac{1}{2\max(\alpha,\mu)}\big(\|\hat{\mathbf{x}}_1\|^2-\|\hat{\mathbf{x}}_{T+1}\|^2\big)+\frac{1}{\max(\alpha,\mu)}\\
&\quad\Big(\|\hat{\mathbf{x}}_{T+1}\|\Big\|\mathbf{x}_T^*\Big\|-\|\hat{\mathbf{x}}_1\|\Big\|\mathbf{x}_1^*\Big\|+\sum_{t=2}^T\|\hat{\mathbf{x}}_t\|\Big(\Big\|\mathbf{x}_{t-1}^*\Big\|-\Big\|\mathbf{x}_t^*\Big\|\Big)\Big)\\
&\overset{(a)}{\leq}\frac{1}{\max(\alpha,\mu)}R^2+\frac{1}{\max(\alpha,\mu)}R\mathcal{V}_{\mathbf{x}_t^*}^T,
\end{aligned}\tag{53}$$

where (a) is due to the assumed $\|\hat{\mathbf{x}}_1\|=0$, the bound of $\hat{\mathbf{x}}$ in Remark 2, which is $\|\hat{\mathbf{x}}\|\leq R$, and the accumulated hindsight optimum variation $\mathcal{V}_{\mathbf{x}_t^*}^T=\sum_{t=2}^T\Big(\Big\|\mathbf{x}_{t-1}^*-\mathbf{x}_t^*\Big\|\Big)$.

$$\begin{aligned}
&\frac{1}{2\max(\alpha,\mu)}\sum_{t=1}^T\Big(\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_t\Big\|^2-\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_{t+1}\Big\|^2\Big)\\
&\leq\frac{1}{2\max(\alpha,\mu)}\Big(\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_1\Big\|^2-\Big\|\boldsymbol{\lambda}_t^*-\hat{\boldsymbol{\lambda}}_{T+1}\Big\|^2\Big)\\
&\leq\frac{1}{2\max(\alpha,\mu)}\Big\|\boldsymbol{\lambda}_t^*\Big\|^2,
\end{aligned}\tag{54}$$

where due to the non-negativity of $\Big\|\boldsymbol{\lambda}_t^*-\boldsymbol{\lambda}_{T+1}\Big\|^2$ and $\Big\|\hat{\boldsymbol{\lambda}}_1\Big\|=0$. Using Lemma 3, we have

$$\begin{aligned}
&\sum_{t=1}^T\mathcal{L}_t^r(\mathbf{x}_t,\boldsymbol{\lambda}_t^*)-\sum_{t=1}^T\mathcal{L}_t^r(\mathbf{x}_t^*,\boldsymbol{\lambda}_t)\\
&\leq\frac{R^2+R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha,\mu)}+\frac{1}{2\max(\alpha,\mu)}\Big\|\boldsymbol{\lambda}_t^*\Big\|^2+\alpha TG^2,
\end{aligned}\tag{55}$$

where $\|\nabla\psi_t(\mathbf{x}_t)\|\leq G$ in Remark 1. Expanding the left hand side of (55) using (13), we get

$$\begin{aligned}
&\sum_{t=1}^T\psi_t(\mathbf{x}_t)+\sum_{t=1}^T\boldsymbol{\lambda}_t^{*\top}\mathbf{g}_t(\mathbf{x}_t)-\sum_{t=1}^T\frac{\delta\alpha}{2}\Big\|\boldsymbol{\lambda}_t^*\Big\|^2-\sum_{t=1}^T\psi_t(\mathbf{x}_t^*)\\
&\qquad-\sum_{t=1}^T\boldsymbol{\lambda}_t{}^{\top}\mathbf{g}_t(\mathbf{x}_t^*)+\sum_{t=1}^T\frac{\delta\alpha}{2}\|\boldsymbol{\lambda}_t\|^2\\
&\leq\frac{R^2+R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha,\mu)}+\frac{1}{2\max(\alpha,\mu)}\Big\|\boldsymbol{\lambda}_t^*\Big\|^2+\alpha TG^2.
\end{aligned}\tag{56}$$

Since $\mathbf{g}_t(\mathbf{x}_t^*)\leq\mathbf{0}$, $\boldsymbol{\lambda}_t\geq\mathbf{0},\forall t$, and δ as well as α are positive real numbers, we can write (56) as

$$\begin{aligned}
&\sum_{t=1}^T\psi_t(\mathbf{x}_t)+\sum_{t=1}^T\boldsymbol{\lambda}_t^{*\top}\mathbf{g}_t(\mathbf{x}_t)-\sum_{t=1}^T\frac{\delta\alpha}{2}\Big\|\boldsymbol{\lambda}_t^*\Big\|^2-\sum_{t=1}^T\psi_t(\mathbf{x}_t^*)\\
&\qquad-\frac{1}{2\max(\alpha,\mu)}\Big\|\boldsymbol{\lambda}_t^*\Big\|^2\\
&\leq\frac{R^2+R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha,\mu)}+\alpha TG^2.
\end{aligned}\tag{57}$$

There exists $\boldsymbol{\lambda}_t^*=\boldsymbol{\lambda}^*=\frac{\sum_{t=1}^T\mathbf{g}_t(\mathbf{x}_t)}{\delta\alpha T+1/\max(\alpha,\mu)},\forall t$ that maximizes the left hand side of (57) and can be found by setting the derivative of $-\Big(\frac{\delta\alpha T}{2}+\frac{1}{2\max(\alpha,\mu)}\Big)\|\boldsymbol{\lambda}^*\|^2+\sum_{t=1}^T\boldsymbol{\lambda}^{*\top}\mathbf{g}_t(\mathbf{x}_t)$ with respect to $\boldsymbol{\lambda}^*$ equal to zero. Substituting $\boldsymbol{\lambda}_t^*=\frac{\sum_{t=1}^T\mathbf{g}_t(\mathbf{x}_t)}{\delta\alpha T+1/\max(\alpha,\mu)},\forall t$ in (57), we arrive at

$$\begin{aligned}
&\sum_{t=1}^T\psi_t(\mathbf{x}_t)-\sum_{t=1}^T\psi_t(\mathbf{x}_t^*)+\frac{\Big\|\sum_{t=1}^T\mathbf{g}_t(\mathbf{x}_t)\Big\|^2}{2(\delta\alpha T+\frac{1}{\max(\alpha,\mu)})}\\
&\leq\frac{R^2+R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha,\mu)}+\alpha TG^2.
\end{aligned}\tag{58}$$

Since the third term on the left hand side of (58) is positive, the upper bound of the dynamic regret is given by

$$\mathrm{R}_T\overset{(a)}{=}\sum_{t=1}^T[\psi_t(\mathbf{x}_t)-\psi_t(\mathbf{x}_t^*)]\leq\frac{R^2+R\mathcal{V}_{\mathbf{x}_t^*}^T}{\max(\alpha,\mu)}+\alpha TG^2,\tag{59}$$

where (a) is the definition of the dynamic regret. To find the lower bound of the regret at each time-slot, we use mean-value theorem stating that there exists $\mathbf{x}'$ such that

$$\begin{aligned}\psi_t(\mathbf{x}_t) - \psi_t(\mathbf{x}_t^*) &= \nabla\psi_t(\mathbf{x}')(\mathbf{x}_t - \mathbf{x}_t^*) \\ &\overset{(b)}{\geq} -\left\|\nabla\psi_t(\mathbf{x}')\right\|\left\|\mathbf{x}_t - \mathbf{x}_t^*\right\| \\ &\overset{(c)}{\geq} -RG \end{aligned} \tag{60}$$

where (b) follows from the Cauchy-Schwartz inequality; (b) uses the constants G and R that defined in Remark 1 and 2, respectively. Therefore, the lower bound of the dynamic regret is given by $\mathrm{R}_T \geq -RGT$.

F. Proof for Theorem 2

Referring to (9) that defines the violation of constraints V_T as $\mathrm{V}_T = \left\|\mathcal{P}_{\mathbb{R}_+^{M+N}}\left\{\sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t)\right\}\right\|$, substituting the dynamic regret $\sum_{t=1}^T[\psi_t(\mathbf{x}_t) - \psi_t(\mathbf{x}_t^*)]$ by its lower bound $-RGT$, i.e., using (60), in (58) and rearranging the terms, we get

$$\begin{aligned}\mathrm{V}_T \leq \left\|\sum_{t=1}^T \mathbf{g}_t(\mathbf{x}_t)\right\| \leq \Bigg[\Bigg(\frac{2R^2}{\max(\alpha,\mu)} + \frac{2R^{\prime}V_{\mathbf{x}_t^*}^T}{\max(\alpha,\mu)} + 2\alpha TG^2 \\ + 2RGT\Bigg)\Bigg(\delta\alpha T + \frac{1}{\max(\alpha,\mu)}\Bigg)\Bigg]^{1/2}. \end{aligned} \tag{61}$$

This completes the proof.

REFERENCES

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile Edge Computing: A Survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, Feb 2018.

[2] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[3] Z. Chen *et al.*, "An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*. ACM, 2017, pp. 1–14.

[4] M. Chen and Y. Hao, "Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, Mar 2018.

[5] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal Workload Allocation in Fog-Cloud Computing Towards Balanced Delay and Power Consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, Dec 2016.

[6] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE International Symposium on Information Theory (ISIT)*. IEEE, Jul 2016, pp. 1451–1455.

[7] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, "Computation Offloading and Resource Allocation in Wireless Cellular Networks With Mobile Edge Computing," *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924–4938, Aug 2017.

[8] A. Koppel, F. Y. Jakubiec, and A. Ribeiro, "A Saddle Point Algorithm for Networked Online Convex Optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 19, pp. 5149–5164, Oct 2015.

[9] L. Zhang, T. Yang, R. Jin, and Z.-H. Zhou, "Dynamic regret of strongly adaptive methods," in *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018, pp. 5877–5886.

[10] O. Besbes, Y. Gur, and A. Zeevi, "Non-Stationary Stochastic Optimization," *Operations Research*, vol. 63, no. 5, pp. 1227–1244, Oct 2015.

[11] E. C. Hall and R. M. Willett, "Online Convex Optimization in Dynamic Environments," *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 647–662, Jun 2015.

[12] M. Suhas, *Small Scale Fading in Radio Propagation*, Department of Electrical Engineering, Rugters University, Lecture Notes for Wireless Communication Technologies, Spring, 2005, [Online]. Available: http://www.winlab.rutgers.edu/~suhas/wct2.pdf.

[13] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: John Wiley & Sons, Inc., Sep 2005.

[14] A. Nemirovski, "Prox-Method with Rate of Convergence O (1/ t) for Variational Inequalities with Lipschitz Continuous Monotone Operators and Smooth Convex-Concave Saddle Point Problems," *SIAM Journal on Optimization*, vol. 15, no. 1, pp. 229–251, Jan 2004.

[15] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge: Cambridge University Press, Jun 2004, vol. 25, no. 3.

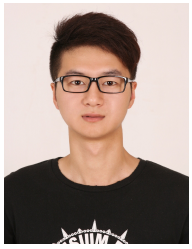
[16] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic Joint Radio and Computational Resource Management for Multi-User Mobile-Edge Computing Systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, Sep 2017.

[17] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, Apr 2016, pp. 1–9.

[18] M. J. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan and Claypool, 2010.

[19] M. Zinkevich, "Online Convex Programming and Generalized Infinitesimal Gradient Ascent," *Proceedings, Twentieth International Conference on Machine Learning*, vol. 2, pp. 928–935, 2003.

[20] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with Bregman Divergences," *Journal of Machine Learining Research*, vol. 6, pp. 1705–1749, 2005.



Zhenfeng Sun (M'16) received the B.Eng in Electronic Engineering from De Montfort University, U.K., in 2016, and the M.S. degree in Electronic Engineering from King's College London, U.K., in 2017. He is currently pursuing the Ph.D. degree with the Centre for Telecommunications Research, Department of Engineering, King's College London, U.K. His research interests include online optimization, reinforcement learning, mobile edge technologies and radio resource management.

Mohammad Reza Nakhai (M'88-SM'07) received the B.Sc. and M.Sc. degrees in electrical engineering from the Sharif University of Technology, Tehran, Iran, in 1984 and 1987, respectively, and the Ph.D. degree in electronic engineering from Kings College London, University of London, U.K., in 2000. From 1988 to 1995, he was with the Sharif University of Technology as a member of Research Faculty and worked on various aspects of signal processing and communications. In 2000, he joined the Centre for Communication Systems Research, University of Surrey, U.K., as a Post-Doctoral Research Fellow. Then, he joined the Department of Electronic Engineering, Kings College London, in 2001, as a member of Academic Staff, where he is currently with the Department of Engineering, Centre for Telecommunications Research. His current research interests include machine learning and artificial intelligence for wireless communications applications, wireless network optimization for energy efficiency, game theory and optimization for wireless networks, cognitive radio communications, and signal processing. He currently serves as an Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.